Introdução

Marina Andretta

ICMC-USP

4 de agosto de 2014

Baseado no livro Numerical Optimization, de J. Nocedal e S. J. Wright.

Otimização

Otimizar significa encontrar a melhor maneira de fazer algo, dada uma medida do que é ser "melhor".

Estamos sempre otimizando:

- Quando fazemos compras, queremos minimizar o dinheiro gasto, ou maximizar a qualidade do que foi comprado;
- Quando fazemos matrícula, queremos fazer o maior número possível de disciplinas, sem, no entanto, prejudicar nosso desempenho;
- Quando organizamos as horas de estudo, queremos aprender o máximo possível, de preferência no menor tempo.

Formulação matemática

Matematicamente falando, otimizar significa maximizar ou minimizar uma função sujeita a restrições a suas variáveis. Usaremos a seguinte notação:

- x é um vetor de variáveis, também chamadas de incógnitas ou parâmetros;
- f é a função objetivo, a função de x que deve ser minimizada ou maximizada;
- c é um vetor de restrições que o ponto x deve satisfazer. Este é um vetor de funções de x. O número de componentes de c é o número de restrições em x.

Formulação matemática

Assim, podemos escrever problemas de otimização da seguinte maneira:

$$\begin{array}{lll} \text{Minimizar} & f(x) & \\ \text{sujeita a} & c_i(x) = 0, & i \in \mathcal{E} \\ & c_i(x) \geq 0, & i \in \mathcal{I} \end{array} \tag{1}$$

onde

- $x \in \boldsymbol{R}^n$, $f \in \boldsymbol{R}^n \to \boldsymbol{R}$, $c_i \in \boldsymbol{R}^n \to \boldsymbol{R}$,
- $\mathcal{E}$ e $\mathcal{I}$ são conjuntos de índices.

Exemplo

$$\begin{array}{ll} \text{Minimizar} & (x_1-2)^2+(x_2-1)^2 \\ \text{sujeita a} & x_1^2-x_2\leq 0, \\ & x_1+x_2\leq 2. \end{array}$$

Este problema pode ser escrito da seguinte forma:

$$\text{Minimizar} \quad f(x)=(x_1-2)^2+(x_2-1)^2, x=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

$$\text{sujeita a} \quad c(x)=\begin{bmatrix} c_1(x) \\ c_2(x) \end{bmatrix}=\begin{bmatrix} -x_1^2+x_2 \\ -x_1-x_2+2 \end{bmatrix}\geq\begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$\mathcal{I}=\{1,2\}, \mathcal{E}=\emptyset.$$

Exemplo

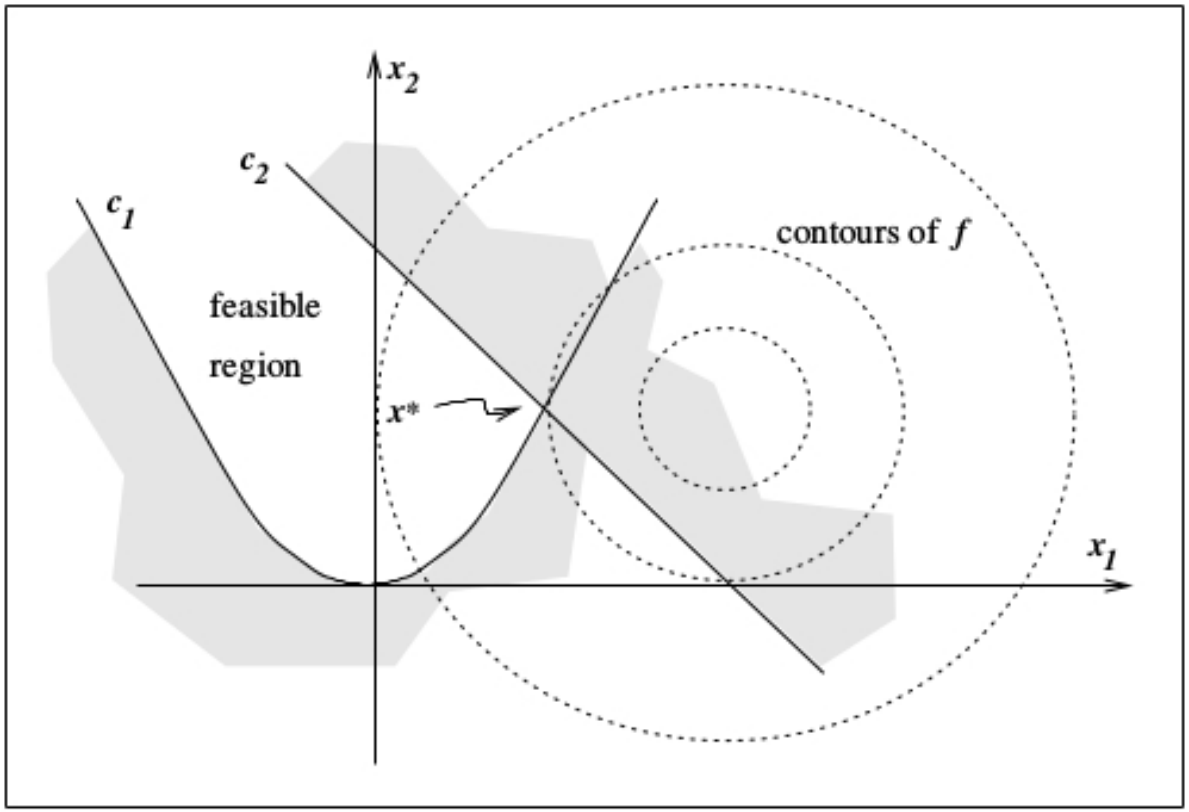

Figura: Exemplo de problema de otimização (Figura 1.1 de Numerical Optimization, de J. Nocedal e S. J. Wright)

Formulação matemática

Note que um problema de maximizar f pode ser substituído por um problema de minimizar $-f$.

Problemas de otimização em geral podem ser escritos usando a formulação (1), que chamaremos de formulação padrão.

Exemplo: problema de transporte

Uma empresa química tem 2 fábricas F_1 e F_2 e 12 clientes R_1, R_2, ..., R_{12}. Cada fábrica F_i pode produzir a_i toneladas de um certo produto químico por semana; a_i é chamado capacidade da planta. Cada cliente R_j possui uma demanda conhecida b_j de toneladas de um produto por semana. O custo de enviar uma tonelada de produto da fábrica F_i para o cliente R_j é c_{ij}.

O problema é determinar quanto do produto enviar de cada fábrica para cada cliente de modo a satisfazer todas as restrições e minimizar o custo. As variáveis do problema são x_{ij}, $i = 1, 2$, $j = 1, ..., 12$, onde x_{ij} é o número de toneladas do produto enviado da fábrica F_i ao cliente R_j.

Exemplo: problema de transporte

$$
\begin{aligned}
\text{Minimizar} \quad & \textstyle\sum_{ij} c_{ij} x_{ij} \\
\text{sujeita a} \quad & \textstyle\sum_{j=1}^{12} x_{ij} \leq a_i, \quad i = 1, 2, \\
& \textstyle\sum_{i=1}^{2} x_{ij} \geq b_j, \quad j = 1, ..., 12, \\
& x_{ij} \geq 0, \qquad\qquad i = 1, 2, \quad j = 1, ..., 12,
\end{aligned}
$$

Otimização contínua *versus* discreta

Em alguns casos, o valor das variáveis pode somente assumir valores inteiros.

Suponha que a empresa do exemplo anterior fabricasse pratos, no lugar de produtos químicos. Neste caso, a quantidade de produto a ser enviada das fábricas para os clientes (x_{ij}) deveria ser inteira. Ou seja, deveria haver uma restrição do tipo $x_{ij} \in Z$, para todo i e j. Este problema seria um problema de programação inteira.

Otimização contínua *versus* discreta

O termo genérico otimização discreta se refere a problemas cujas variáveis assumem valores em um conjunto finito. Em contraste, o termo otimização contínua se refere a problemas cujas variáveis podem assumir valores em conjuntos infinitos não-enumeráveis, tipicamente, um conjunto de valores reais.

Problemas de otimização contínua normalmente são mais fáceis de resolver do que problemas de otimização discreta. A suavidade da função objetivo e das funções das restrições permite que, a partir da informação em um ponto x, possamos deduzir o comportamento destas funções em uma vizinhança de x. Métodos para resolver problemas de otimização contínua fazem uso destas propriedades quando buscam a solução do problema.

Otimização contínua *versus* discreta

No caso de problemas de otimização discreta, uma solução "vizinha" de outra pode não apresentar valores da função objetivo próximos. Neste caso, uma alternativa seria enumerar as possíveis soluções, buscando a de menor valor de função objetivo. No entanto, para problemas maiores, o tempo gasto com essa busca é proibitivo.

Em alguns casos, algumas variáveis do problema são contínuas e outras são discretas. Estes problemas são chamados de problemas de programação inteira mista.

Otimização irrestrita *versus* restrita

Problemas do tipo (1) podem ser classificados de várias maneiras:

- Quanto à natureza da função objetivo: linear, não-linear, convexa, etc;
- Quanto ao número de variáveis: pequeno, médio ou grande;
- Quanto à suavidade das funções: diferenciável ou não-diferenciável;
- Etc.

Otimização irrestrita *versus* restrita

Uma distinção importante entre problemas do tipo (1) é com relação às restrições. Se não há restrições, dizemos que o problema é irrestrito. Se há pelo menos uma restrição, dizemos que o problema é restrito.

Problemas irrestritos surgem de modelos de problemas que naturalmente não possuem restrições. Em alguns casos, o problema teria restrições, mas elas podem ser ignoradas. Em outros casos, problemas originalmente restritos têm as restrições transformadas em um termo de penalidade na função objetivo, tornando-se irrestritos.

Otimização irrestrita *versus* restrita

Problemas restritos surgem de modelos que possuem restrições explícitas. As restrições podem ser:

- Simples limitantes nas variáveis, como $0 \leq x \leq 10$. Estas restrições são chamadas de restrições de caixa;
- Restrições lineares, como $\sum_i x_i \leq 1$;
- Restrições não-lineares, como $\sin(x) + \cos(x) = 1$.

Problemas nos quais tanto a função objetivo como as restrições são lineares em x, são chamados de problemas de programação linear. Se a função objetivo ou alguma das restrições não é linear, o problema é chamado de problema de programação não-linear.

Otimização local *versus* global

Os termos otimização local e otimização global se referem à qualidade da solução a ser buscada.

Os algoritmos mais rápidos de otimização buscam encontrar uma solução local, ou seja, um ponto no qual o valor da função objetivo vale menos (no caso de minimização) do que os pontos em sua vizinhança.

Um algoritmo para otimização global se compromete a encontrar um ponto x, dentre os possíveis valores que x pode assumir, que possui o menor valor de função objetivo possível. Em geral, encontrar uma solução global é muito mais difícil do que encontrar uma solução global.

Algoritmos para otimização

Algoritmos de otimização são iterativos. Eles partem de um "chute" inicial dos valores das variáveis e geram uma sequência com valores aprimorados das variáveis até atingir a solução.

A estratégia usada para ir de um iterando a outro é o que distingue um algoritmo de outro. A maioria das estratégias usa o valor da função objetivo f, o valor das restrições c e, possivelmente, a primeira e segunda derivadas dessas funções.

Alguns algoritmos armazenam as informações obtidas em cada iteração, enquanto outros usam apenas informação local do ponto x atual.

Algoritmos para otimização

Todo bom algoritmo de otimização deve possuir os seguintes objetivos:

- Robustez: deve funcionar bem em uma grande variedade de problemas da classe que ele pretende resolver, para todas as escolhas razoáveis de pontos iniciais;
- Eficiência: não deve exigir muito tempo computacional ou espaço em memória para ser executado;
- Precisão: deve ser capaz de identificar uma solução com precisão, sem ser muito sensível a erros nos dados ou a erros de arredondamento que podem ocorrer durante sua execução.

Algoritmos para otimização

Estes objetivos podem ser conflitantes. Por exemplo, um método que convirja rapidamente para a solução pode necessitar de muito espaço de armazenamento para problemas de grande porte. Por outro lado, métodos mais robustos podem ser mais lentos.

O equilíbrio entre velocidade de convergência e armazenamento necessário, robustez e velocidade, etc, são pontos centrais em otimização numérica.